PROGRAMACIÓN EN ENTORNOS VIRTUALES

MEMORY

Índice

Introducción

Objetivos

En esta práctica se han desarrollado dos programas en C++ utilizando Qt Designer, centrándose en el diseño de interfaces interactivas y la implementación de lógica de juego y procesamiento de imágenes. El programa es una versión del conocido juego Memory, este ejercicio refuerza los conocimientos en programación orientada a eventos y manipulación de interfaces gráficas.

- **Memory**: Implementar un juego de memoria con una interfaz gráfica interactiva, sonidos asociados a diferentes acciones y una lógica de juego que maneje emparejamientos de iconos.

Herramientas de Programación Utilizadas

En el desarrollo del ejercicio de la práctica se utilizaron diversas herramientas de programación que permitieron la creación eficiente de la aplicación, así como la implementación de funcionalidades específicas. A continuación, se describen las principales herramientas y tecnologías empleadas:

Herramientas archivos .h/.cpp

QsoundEffect

Clase de Qt para manejar efectos de sonido, utilizada para reproducir sonidos en el juego de memoria, mejorando la interacción del usuario.

```
openSound.setSource(QUrl::fromLocalFile(":/sonidos/open.wav"));
doneSound.setSource(QUrl::fromLocalFile(":/sonidos/done.wav"));
closeSound.setSource(QUrl::fromLocalFile(":/sonidos/close.wav"));
errorSound.setSource(QUrl::fromLocalFile(":/sonidos/error.wav"));
winSound.setSource(QUrl::fromLocalFile(":/sonidos/win.wav"));
```

En el juego de memoria, **QSoundEffect** se utilizó para proporcionar retroalimentación auditiva al usuario cuando se descubre una casilla, se empareja una casilla, se cierra una casilla, se comete un error y al ganar el juego.

QMessageBox

Clase utilizada para mostrar diálogos informativos y de confirmación al usuario, útil para alertas y mensajes de información.

```
QMessageBox::about(this, "Acerca de", "Programa para mejorar la memoria, creado por Carlos Rovira López");
```

En ambas prácticas, **QMessageBox** se utilizó para proporcionar mensajes de información y confirmación, como en el caso de un usuario que gana el juego de memoria, implementación de “Acerca de”.

QTimer

Clase que permite ejecutar código después de un intervalo de tiempo, utilizada en el juego de memoria para manejar retrasos al verificar coincidencias.

```
QTimer::singleShot(1000, this, &MainWindow::checkForMatch);
```

En el juego de memoria, **QTimer** se utilizó para proporcionar un retraso antes de verificar si dos imágenes coinciden, lo que permite al usuario ver ambas imágenes antes de que se oculten nuevamente si no coinciden.

Herramientas archivos .ui

Los archivos **.ui** se utilizan para diseñar la interfaz de usuario de forma visual en Qt Designer. Contienen la disposición de los widgets y sus propiedades.

MainWindow (ejercicio 1: Juego de Memoria)

Elementos:

1. **centralwidget (QWidget):** Contiene todos los elementos de la interfaz principal.
2. **horizontalLayout (QHBoxLayout):** Organiza los elementos de forma horizontal.
3. **gridLayout_2 (QGridLayout):** Disposición en cuadrícula para los botones del juego.
4. **horizontalSpacer y verticalSpacer (QSpacerItem):** Añaden espacios vacíos para centrar los elementos.
5. **menubar (QMenuBar):** Barra de menú superior con opciones.
    - **menuAcerca (QMenu):** Menú desplegable con información sobre la aplicación.
        - **actionAcercade (QAction):** Acción para mostrar el cuadro de diálogo "Acerca de".
    - **menujuego (QMenu):** Menú desplegable con opciones de juego.
        - **actionreiniciar (QAction):** Acción para reiniciar el juego.
        - **actionsalir (QAction):** Acción para salir de la aplicación.
6. **statusbar (QStatusBar):** Barra de estado en la parte inferior de la ventana.
7. **toolBar (QToolBar):** Barra de herramientas con acciones rápidas.
    - **actionAcercade, actionreiniciar, actionsalir (QAction):** Acciones correspondientes en la barra de herramientas.

Uso dado en la práctica:

QWidget: Actúa como el contenedor principal de los elementos.

QGridLayout: Disposición de los botones en una cuadrícula para representar el tablero de juego.

QSpacerItem: Centran el tablero en la ventana.

QMenuBar y QToolBar: Proporcionan acceso a las funciones principales del juego (reiniciar, salir, información).

Código Fuente

El código fuente desarrollado para el Juego de Memoria en Qt y C++ demuestra una implementación avanzada y detallada de técnicas de programación orientadas a la creación de aplicaciones interactivas. La clase principal `MainWindow` hereda de `QMainWindow` y sirve como contenedor para la interfaz gráfica y la lógica del juego. Los elementos de la interfaz, definidos en el archivo `mainwindow.ui`, incluyen una matriz de botones que representan las cartas del juego. Cada botón está asociado a un icono que se baraja al inicio de cada partida para garantizar la aleatoriedad.

La gestión de eventos de clic es central en la funcionalidad del juego. Al pulsar un botón, se verifica si es la primera o segunda selección y se muestra el icono correspondiente. Si se seleccionan dos cartas, el método `checkForMatch` comprueba si los iconos coinciden. En caso de coincidencia, se actualizan los iconos y se reproducen efectos de sonido utilizando `QSoundEffect`, proporcionando una retroalimentación auditiva inmediata al usuario. Los sonidos se configuran para diferentes acciones, como descubrir una casilla, emparejar una casilla, cerrar una casilla y errores. Esta funcionalidad no solo mejora la experiencia del usuario sino que también incorpora elementos de accesibilidad.

La verificación de victoria se realiza mediante el método `checkIfGameWon`, que comprueba si todos los botones han sido emparejados correctamente. Si se cumplen las condiciones de victoria, se muestra un mensaje de felicitación y se ofrece al usuario la opción de reiniciar el juego o salir de la aplicación. Este enfoque modular y orientado a objetos garantiza que la lógica del juego esté claramente separada de la presentación, facilitando futuras extensiones y mantenimientos del código.

Errores y Tecnologías

Durante el desarrollo de la práctica, enfrentamos varios retos técnicos significativos:

Uno de los principales problemas fue la lógica para realizar el match de dos botones. Inicialmente, el programa no distinguía correctamente entre los botones ya emparejados y los nuevos intentos, lo que resultaba en fallos de comparación y emparejamientos incorrectos. Para resolver esto, implementamos propiedades adicionales en los botones, como `firstButton` y `secondButton`, que almacenan temporalmente los botones seleccionados. Además, añadimos propiedades como `matched` para marcar los botones que ya habían sido emparejados correctamente. Se utilizó un temporizador (`QTimer`) para retrasar la comparación y permitir al usuario ver ambas imágenes antes de que se oculten nuevamente si no coincidían. Finalmente, se incorporaron efectos de sonido para proporcionar retroalimentación auditiva en cada acción, lo que mejoró significativamente la experiencia del usuario.

Experimentación y validación

En la fase de experimentación, se llevaron a cabo pruebas concretas para evaluar la robustez y funcionalidad del programa. Algunos experimentos incluyeron:

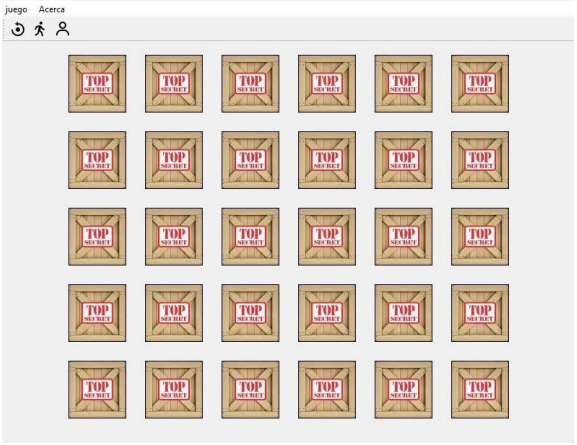
MEMORY

Pruebas de Funcionalidad Básica

Prueba de Inicialización del Juego:

Acción: Ejecutar el programa.

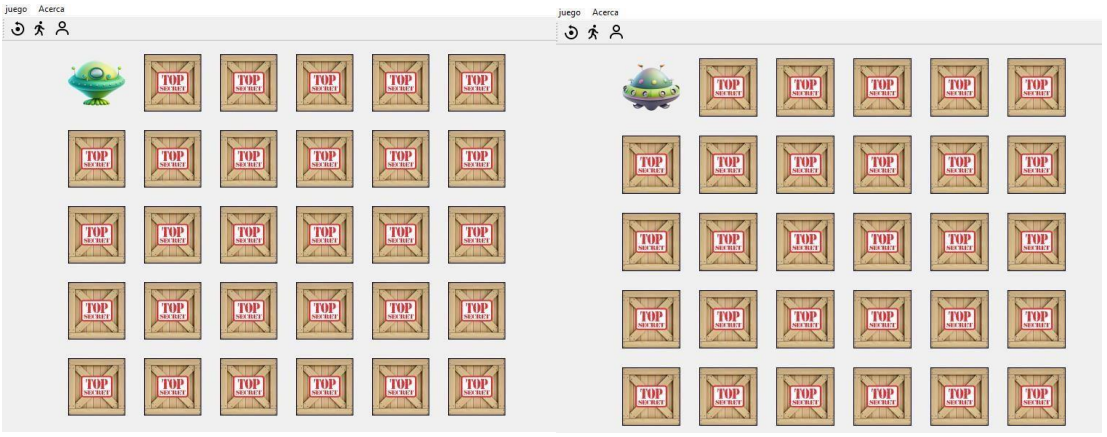
Verificación: Todas las celdas deben estar ocultas con el icono topsecret.png. No debe haber celdas descubiertas.



Prueba de Carga de Iconos:

Acción: Reiniciar el juego varias veces.

Verificación: Verificar que los iconos se barajan correctamente y que los pares de iconos están distribuidos aleatoriamente en cada reinicio**.**



Prueba de Emparejamiento Correcto:

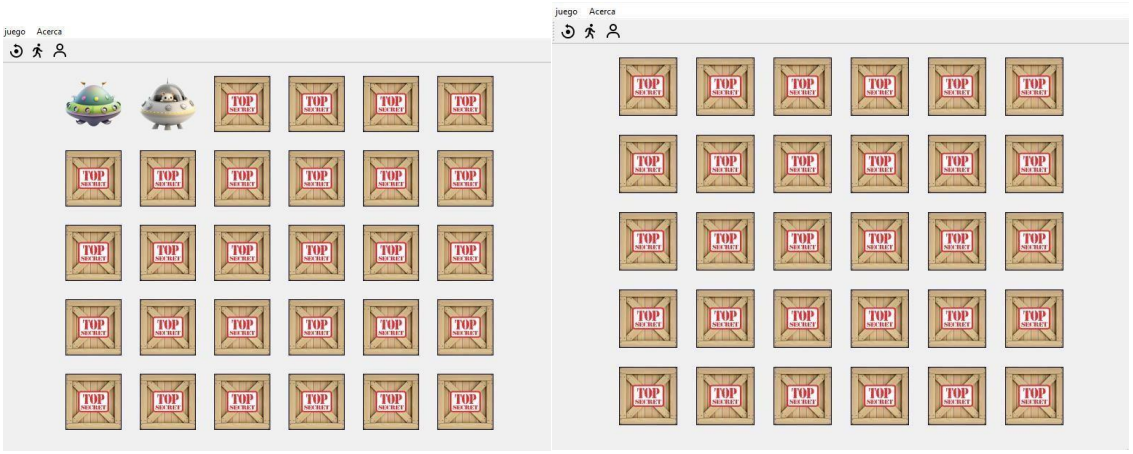
Acción: Hacer clic en dos celdas con el mismo icono.

Verificación: Las celdas deben quedar validadas mostrando el icono con la marca (ufoXXc.png) y reproducir el sonido done.wav. Las celdas deben deshabilitarse.

Prueba de No Emparejamiento:

Acción: Hacer clic en dos celdas con diferentes iconos.

Verificación: Las celdas deben volver a ocultarse con el icono topsecret.png y reproducir el sonido close.wav.

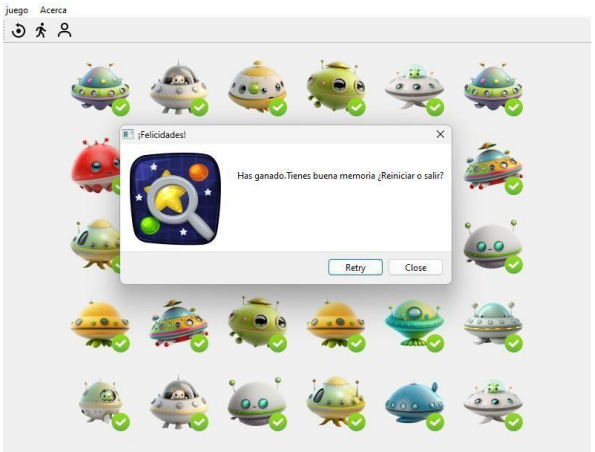


Pruebas de Finalización del Juego

Prueba de Victoria:

Acción: Emparejar todas las celdas correctamente.

Verificación: Al emparejar todas las celdas, debe mostrarse un mensaje de felicitación y reproducir el sonido win.wav. El mensaje debe ofrecer opciones para reiniciar o salir.

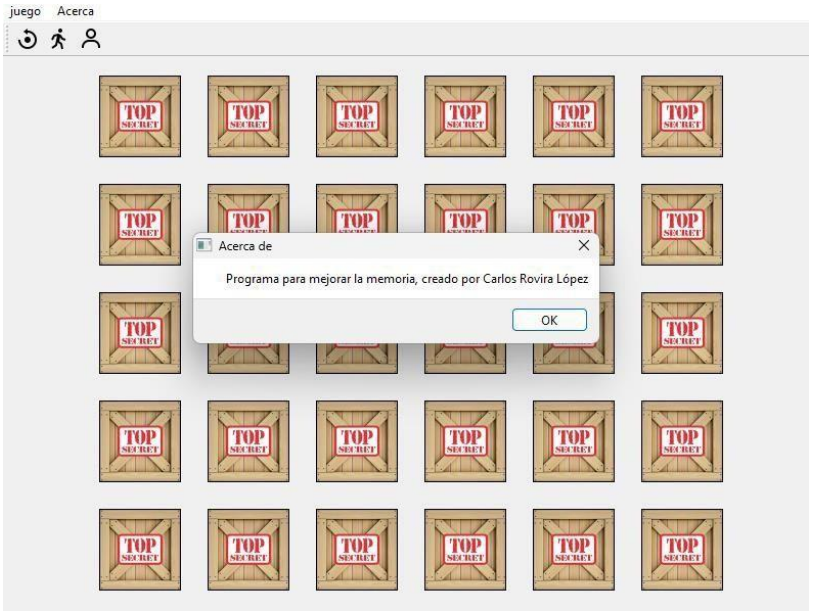


Pruebas de Interfaz de Usuario

Prueba de Menú "Acerca de":

Acción: Seleccionar la opción "Acerca de" en el menú.

Verificación: Debe mostrarse un cuadro de diálogo con información sobre el autor del programa.



Prueba de Estabilidad a Largo Plazo:

Acción: Dejar el juego en ejecución durante un periodo prolongado (varias horas).

Verificación: Verificar que el juego sigue respondiendo correctamente y no se producen errores ni fugas de memoria.

Pruebas en casos extremos

Prueba de Celdas No Interactuables:

Acción: Intentar interactuar con celdas ya validadas (emparejadas).

Verificación: Las celdas ya emparejadas no deben responder a los clics y deben permanecer visibles con el icono de validación (ufoXXc.png)